成績資料庫

初學者的統計 (I): 列表與迴圈

複習資料型態

- 目前已學習到的資料型態
  - 整數 3
  - 小數 3.14
  - 字串 'NTNU'

```
print(type(3))
print(type(3.14))
print(type('NTNU'))
a = 3
print(type(a))
a = 3.14
print(type(a))
```

```
<class 'int'>
<class 'float'>
<class 'str'>
<class 'int'>
<class 'float'>
```

type() 函式可以取得資料的型態。

列表 (list) 降臨

- 不同的資料型態有不同的運算/操作
  - 記得 3*2 和 '3'*3 的差別嗎？
  - 除了 + 和 * 之外，字串還有很多操作，例如 split()。

split 劈開

```
s = 'NTNU is great!'
words = s.split()
print(type(words))
print(words)
```

```
<class 'list'>
['NTNU', 'is', 'great!']
```

  - s.split() 代表對變數 s 施行 split() 操作。
    - 不是任何資料型態都有 split()。
  - 從執行結果，我們推敲出
    - 對字串進行 split() 操作會得到一種 list 型態的資料。
    - 對字串進行 split() 操作，會以空白字元分隔出多個字串，並以 list 型態儲存之。

列表 (list) 降臨

發揮實驗精神

```
print('NTNU      is great!'.split())
print('abcde'.split())
print('123, 456, 789'.split())
print('123,456,789'.split())
```

```
['NTNU', 'is', 'great!']
['abcde']
['123,', '456,', '789']
['123,456,789']
```

- 我們推敲：
  - split() 以空白字元分隔出來的字串中不會含有空白字元。
  - 如果原字串中沒有空白字元，split() 還是會產生只有單一字串的列表。

split() 可以傳入自訂的分隔字元或字串，這部份就先留給有興趣的同學自行實驗囉！

例：

```
'1,2,3,4'.split(',')
'there--are--many--words'.split('--')
```

取得列表中的資料

- 從字串中分離出個別資料，成為一個字串的列表後，我們該如何取得指定的資料呢？
- 列表型態支援下標運算子 [ ]，以從 0 開始的數字取用列表中的個別資料（往後稱之為「元素」element)。

```
s = '2018 9 26'
num = s.split()
print(num)
print(num[0])
print(num[0]+num[1])
print(int(num[0])+int(num[1]))
```

```
['2018', '9', '26']
2018
20189
2027
```

取得列表中的資料

- 索引值的實驗

```
s = '2018 9 26'
num = s.split()
print(num)
print(num[0])
#print(num[3])          #IndexError: list index out of range
#print(num['first'])    #TypeError: list indices must be integers or slices, not str
#print(num[1.5])        #TypeError: list indices must be integers or slices, not float
print(num[-1])
print(num[-3])
#print(num[-4])         #IndexError: list index out of range
```

```
['2018', '9', '26']
2018
26
2018
```

取得列表中的資料

- 我們學會了
  - 字串支援 split() 操作，而 split() 會產生一個字串的列表。
  - 列表是種可存放一個以上資料的容器。
  - 以從 0 開始的索引值取用列表中的單一元素。

- 如何寫一個程式，一次讀入三個整數值，然後輸出它們的總和？

```
num = input('請輸入三個整數> ').split()
ans = int(num[0])+int(num[1])+int(num[2])
print(ans)
```

還不錯，可是 ……

- 如果有很多個數怎麼辦？
- 如果不知道有幾個數怎麼辦？

取得列表中的資料：for 句型

依序取出列表中所有元素

- for 句型可以依序取出列表中所有元素，例如：

```
num = input('請輸入任意個整數> ').split()
ans = 0
for e in num:
    ans = ans + int(e)
print(ans)
```

```
請輸入任意個整數> 1 2 9
12
```

```
請輸入任意個整數> 1 3 4 5
13
```

- 執行細節
  - 如果使用者輸入 '1 2 9'，split() 產生列表 ['1', '2,' '9']。
  - for 句型會依序令變數 e 指稱 '1', '2', 和 '9'，對 ans 和 e 執行 ans = ans + int(e)。

如果第四行程式寫成 ans = ans + e 會怎麼樣呢？
如果再把第二行程式寫成 ans = '' 會怎麼樣呢？

取得列表中的資料：for 句型

for 句型

- 中文

對列表……中所有元素，令……代表之，執行……。

- 英文

For every element in the list …, let … denote it and do … .

- Python 文

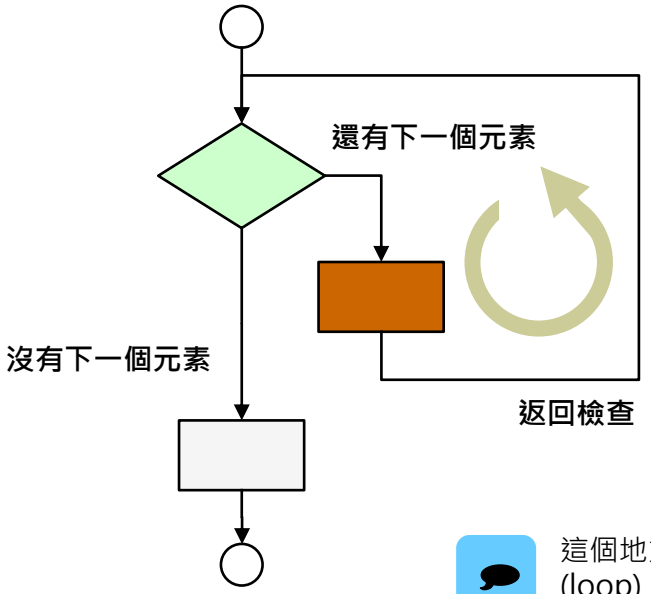
```
for 變數 in 列表:
    動作1
    動作2
非重覆動作
```

```
for e in num:
    ans = ans + int(e)
print(ans)
```

取得列表中的資料：for 句型

for 句型

```
for 變數 in 列表:
    動作1
    動作2
非重覆動作
```



這個地方形成一個迴路(迴圈)(loop)，也因此 for 句型被稱為是一種迴圈控制或者重覆控制。

Snakify 線上練習時間

- 請完成 Snakify 平台主題 **7 Lists** 的習題2 **Even elements**。

參考答案

```
data = input().split()
for e in data:
    if int(e)%2==0:
        print(e)
```

任務：成績處理練習 I

計算平均分數

- 運用所學，修改以下程式，使其具備以下功能：
  - 在一行輸入任意個整數分數
  - 只採計介於 0-100 分的分數
  - 計算並輸出分數的個數，總和以及平均
  - 如果沒有任何合法分數，個數、總和與平均皆為 0。
- 思考程式流程

目標	程式
在一行輸入任意個整數分數	• 使用 `input()` 與 `split()` 得到資料列表
只採計介於 0-100 分的分數	• 使用 `if` 句型判斷數值是否符合
計算分數的個數，總和以及平均	• 使用 `for` 句型檢視列表中的資料 • 使用 `int()` 將字串轉換成整數值 • 使用兩個變數累計資料個數和總和
輸出分數的個數，總和以及平均	• 使用 `print()` 列印結果 • 使用 `if` 句型判斷合法分數個數是否大於零。

任務：成績處理練習 I

計算平均分數

- 運用所學，修改以下程式，使其具備以下功能：
  - 在一行輸入任意個整數分數
  - 只採計介於 0-100 分的分數
  - 計算並輸出分數的個數，總和以及平均
  - 如果沒有任何合法分數，個數、總和與平均皆為 0。

```
num = input('請輸入任意個整數> ').split()
total = 0
count = 0
for e in num:
    total = total + int(e)
    count = count + 1

print('個數=', count, '，總和=', total, '，平均=', total/count)
```

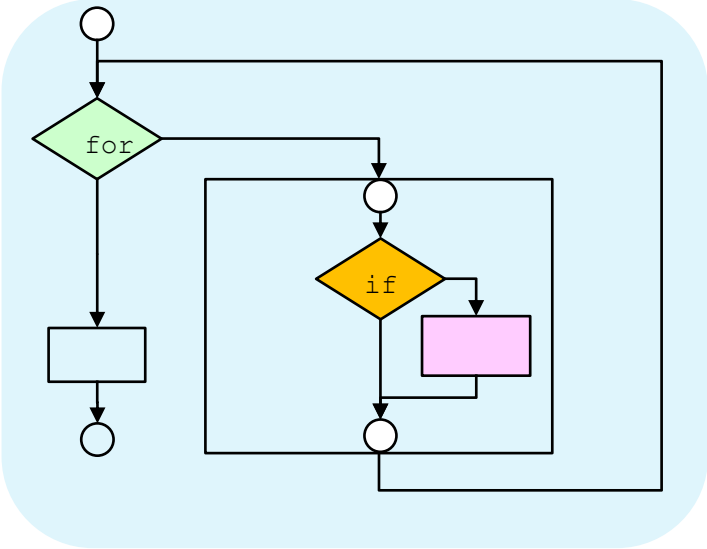
注意：這個程式尚未完成所有功能。

任務：成績處理練習 I

計算平均分數

```
num = input('請輸入任意個整數> ').split()
total = 0
count = 0
for e in num:
    score = int(e)
    if 0<=score<=100:
        total = total + score
        count = count + 1

if count>0:
    print('個數=', count, '，總和=', total, '，平均=', total/count)
else:
    print('個數=0，總和=0，平均=0')
```



為什麼是外層 for 內層 if，不是外層 if 內層 for？
for 句型代表重覆，if 句型代表過濾；我們這裡要對所有成績重覆進行過濾，所以是外層 for 內層 if。

你測試過什麼樣的資料？把你的程式給旁邊的同學測試看看。

成就達成！

恭喜你！完成這個程式，你已具備良好的程式設計能力！

能力	具體作法
向使用者取得大量數據	input()
以易處理的方式儲存數據	字串 split() 與列表 list
以簡潔的程式檢視數據	for 句型
對數據進行檢核與分類	if 句型，關係運算，邏輯運算
（根據分類）進行運算	算術運算
顯示結果	print()

為自己喝采，掌聲鼓勵！！

Snakify 線上練習時間

- 請完成 Snakify 平台主題
  7 Lists 的習題6 **The largest element**。

data	['3',	'8',	'1',	'2',	'9',	'5']
求總和 total	3	11	12	14	23	28

```
for e in data:
    total = total+int(e)
```

data	['3',	'8',	'1',	'2',	'9',	'5']
求最大 largest	3	8	8	8	9	9

```
for e in data:
    [                    ]
```

這裡要作什麼事情，可以得到正確的 largest 數值？

Snakify 線上練習時間

- 請完成 Snakify 平台主題
  7 Lists 的習題6 **The largest element**。

參考答案

```
data = input().split()
i = 0
for e in data:
    e = int(e)
    i += 1
```

利用 i 記下索引 (index) 值

```
data = input().split()
i = 0
largest = int(data[0])
for e in data:
    e = int(e)
    if          :
        largest = e
    i += 1
print(largest)
```

判斷何時該更新 i 和 largest

你的程式

以同樣手法記下 largest_index

Snakify 線上練習時間

請完成 Snakify 平台主題

7 Lists 的習題6 The largest element。

Q1: 下面的程式為什麼不對？

```
data = input().split()
index = 0
largest = int(data[0])
for e in data:
    e = int(e)
    if e>largest:
        largest = e
        index += 1
print(largest, index)
```

在這個時間點觀察變數值

測試案例一

輸入	1	2	3	2	1
largest	1	2	3	3	3
index	0	1	2	2	2

測試案例二（正解應為 4 3，不是 4 2）

輸入	1	3	2	4	1
largest	1	3	3	4	4
index	0	1	1	2	2

歡迎使用 Python Tutor 觀察過程
https://goo.gl/1D7ity

index 記錄的是 e>largest 發生了幾次，
而非 largest 的位置（索引值）。

Snakify 線上練習時間

請完成 Snakify 平台主題

7 Lists 的習題6 **The largest element**。

Q2: 下面的程式為什麼不對？

```
data = input().split()
index = 0
largest = int(data[0])
largest_index = 0
for e in data:
    e = int(e)
    index += 1
    ————————————————————
    if e>largest:
        largest = e
        largest_index = index

print(largest, largest_index)
```

在這個時間點觀察變數值

測試案例一（正解應為 4 3，不是 4 4）

輸入	1	3	2	4	1
largest	1	1	3	3	4
index	1	2	3	4	5
laregst_index	0	0	2	2	4

思考看看 index+=1 應該在什麼時候作才正確？

歡迎使用 Python Tutor 觀察過程
https://goo.gl/x9VGjN

成就達成！

- 現在你有能力撰寫程式分析數據，回答下列問題（以全班分數為例）：
  - 簡單查詢：是否有同學考 100 分？
  - 計數：不及格的同學共有幾位？
  - 極值：全班最高分幾分？最低分幾分？
  - 複雜查詢：不及格的最高分是幾分？

成績資料庫

初學者的統計 (II): 自建列表與內建函式

自建列表與內建函式

- 在撰寫「計算平均分數」程式的過程中，我們學習了迴圈控制和選擇控制的整合應用。
  - 這個模式 - 以迴圈搭配選擇來更新變數，是一個資料處理的基本招式，務必要熟悉。
- 在實務上，計算平均與極值、搜尋與排序，通常會透過內建函式來完成。
  - 我們已學過 `print()` 和 `input()` 兩個函式。
  - 你可以預期，Python 應該內建一些函式讓我們這樣用：

```
avg = sum(numbers) / count(numbers)
minimum = min(numbers)
```

在 Python 中，`count()` 的名字叫作 `len()`。

自建列表與內建函式

實驗時間

```
num = input('請輸入任意個整數> ').split()
#a = total(num)          #NameError: name 'total' is not defined
#a = sum(num)            #TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

- 確實有個內建函式 sum()，但它接受的資料是整數 (int) 而非字串 (str)
  - 別忘了，上述程式中，num 是一個字串的列表。

自建列表與內建函式

實驗時間

```
num = input('請輸入任意個整數> ').split()
minimum = min(num)
print(minimum)
```

```
請輸入任意個整數> 78 9
78
```

- 確實有個內建函式 `min()`，但由於傳入的 `num` 是字串列表，所以得到 `'78'` 和 `'9'` 當中最小的字串 `'78'`。
- 字串的大小關係是以所謂的字典順序來決定。
  - '7' 小於 '9'，'78' 小於 '9'，'789' 小於 '9'，'721' 小於 '73'。
  - 'big' 小於 'small'，'cat' 小於 'dog'，'bat' 小於 'cat'。

你可以在 Python shell 中鍵入 `'big' < 'small'`，shell 會回應 `True`。
如果輸入 `'Big' < 'big'`，也會回應 `True`。為什麼？

自建列表與內建函式

- 經過實驗，我們發現確實有內建函式 sum() 和 min()。
- 若想要得到分數的總和以及最小值，我們不能傳入字串的列表，必須傳入整數的列表。

該怎麼做呢？

自建列表與內建函式

目標：從字串列表建立整數列表

方法 1-1：從無到有 → 使用列表的 append() 操作

- 列表的 append() 操作可將一個元素加入到列表的尾端。

```
str = input('請輸入任意個整數> ').split()
num = []
for e in str:
    num.append(int(e))
print(num)
print(sum(num))
```

```
請輸入任意個整數> 80 90 70
[80, 90, 70]
240
```

自建列表與內建函式

目標：從字串列表建立整數列表

方法 1-2：從無到有 → 使用列表的 + 運算子

- 列表的 + 運算子可將兩個列表接合產生一個新的列表。

```
str = input('請輸入任意個整數> ').split()
num = []
for e in str:
    num = num + [int(e)]
print(num)
print(sum(num))
```

```
請輸入任意個整數> 80 90 70
[80, 90, 70]
240
```

運算子 + 會產生新的列表，再指派給 num。

自建列表與內建函式

- 目標：從字串列表建立整數列表
  - 列表的 `append()` vs. `+` 運算子

```
a = [1, 2, 3]
b = [4, 5]

a.append(b)
print(a)
print(b)
```

```
[1, 2, 3, [4, 5]]
[4, 5]
```

```
a = [1, 2, 3]
b = [4, 5]

a = a + b # 也可以寫成 a+=b
print(a)
print(b)
```

```
[1, 2, 3, 4, 5]
[4, 5]
```

自建列表與內建函式

▶ 目標：從字串列表建立整數列表

方法 1-3：從無到有 → list comprehension

將列表中的所有元素逐一取出並作運算，以這些運算結果建立另一新列表。

```
[運算結果 for 元素 in 列表 ]
```

範例一

```
a = [1, 12, 43]
b = [e+1 for e in a]

print(a)
print(b)
```

```
[1, 12, 43]
[2, 13, 44]
```

範例二

```
a = [1, 2, 3]
b = [e**2 for e in a]

print(a)
print(b)
```

```
[1, 2, 3]
[1, 4, 9]
```

自建列表與內建函式

目標：從字串列表建立整數列表

方法 1-3：從無到有 → list comprehension

```
num = [int(e) for e in input('請輸入任意個整數> ').split()]
print(num)
print('總和=', sum(num))
```

```
請輸入任意個整數> 80 90 70
[80, 90, 70]
總和= 240
```

本方法雖語法較為複雜，但這是 Python 的普遍作法。
考量很久，最後還是覺得應該教大家這個寫法。覺得太複雜的話，也許拆成兩行會好讀一些。

```
data = input('請輸入任意個整數> ').split()
num = [int(e) for e in data]
print(num)
print('總和=', sum(num))
```

自建列表與內建函式

- 呼～～ 喘一口氣！總結一下我們在作什麼。
  - 首先，我們希望運用 Python 內建的資料處理函式，例如 `sum()` 和 `min()`。
  - 我們需要將字串列表轉換成整數列表，這樣子 `sum()` 和 `min()` 才會如我們預料的工作。
  - 接著，我們學會了三種自建列表的方法：
    - 以列表的 `append()` 操作建立列表。
    - 列表的 + 運算子建立列表。
    - 以 list comprehension 建立列表。

自建列表與內建函式

- 現在，我們可以輕鬆呼叫內建函式囉！

```
num = [int(e) for e in input('請輸入任意個整數> ').split()]

print('總和=', sum(num))
print('最大值=', max(num))
print('最小值=', min(num))
print('平均=', sum(num)/len(num))
```

```
請輸入任意個整數> 80 90 70
總和= 240
最大值= 90
最小值= 70
平均= 80.0
```

- len() 函式不管列表中元素的資料型態，它會回傳列表中的元素個數。

自建列表與內建函式

- 帶有過濾條件來建立列表
  - 使用 if 搭配 append() / +運算子

```
old = [int(e) for e in input().split()]
new = []
for e in old:
    if 0<=e<=100:
        new.append(e)

print(old)
print(new)
```

```
1000 3 -4 98 88
[3, 98, 88]
```

  - 使用 list comprehension

```
data = [int(e) for e in input().split() if 0<=int(e)<=100]
```

自建列表與內建函式

常用的內建數學/統計函式

函式名稱	簡述	函式用法
abs()	絕對值	abs(-3) 回傳 3；abs(-3.14) 回傳 3.14。
round()	四捨五入	round(3.4) 回傳 3；round(3.5) 回傳 4； round(3.44, 1) 回傳 3.4；round(3.45, 1) 回傳 3.5。
len()	資料量	len([]) 回傳 0；len([90, 3, 56]) 回傳 3。
min()	最小值	min([9, 3, 2, 1, 5]) 回傳 1； min(['bac', 'ab']) 回傳 'ab'。
max()	最大值	max([9, 3, 2, 1, 5]) 回傳 9； max(['bac', 'ab']) 回傳 'bac'。
sum()	總和	sum([]) 回傳 0；sum([5, 1, 3]) 回傳 9。
sorted()	排序	a = [8, 1, 3] b = sorted(a) print(a, b) [8, 1, 3] [1, 3, 8]

傳入空列表 [] 到 min() 和 max() 會發生錯誤。

statistics 模組中還有 mean()、median()、stdev()、variance() 等等函式可使用。

任務：成績處理練習 II

- 撰寫一個程式，讀入全班成績，過濾非法分數和離群分數，然後輸出最低分、中位數和最高分。
  - 過濾條件一：只留下介於 0 到 100 分的分數。
  - 過濾條件二：只留下和平均值差距三個標準差之內的分數。
    - 注意：平均值和標準差之計算，必須先過濾掉非法分數。
    - 平均值公式：總和 / 個數
    - 標準差公式：

$$SD = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

  - 思路：
    - 先寫下整個資料處理步驟與流程
      - 大流程寫好後，再慢慢拆解（例如一步一步計算標準差）
    - 再寫下每步驟需要使用到的程式關鍵字
    - 最後再開始寫程式

任務：成績處理練習 II

- 你可以像這樣寫下你的想法

目標	程式
在一行輸入任意個整數分數	• 使用 `input()` 與 `split()` 得到資料列表
輸出分數的個數，總和以及平均	• 使用 `for` 句型檢視列表中的資料 • 使用 `int()` 將字串轉換成整數值 • 使用兩個變數累計資料個數和總和

- 你也可以像這樣畫一個流程圖

https://www.reddit.com/r/funny/comments/jgb12/am_i_a_horse_a_helpful_flowchart/

任務：成績處理練習 II

測試程式

- 測試的時間點依你對程式的掌握度而定。
- 當你覺得程式有點大了，就可以測試該段程式碼的正確性。
  - 例如你可以在寫完過濾條件一的時候，就測試一遍。
- 記得喔！我一直強調學習測試程式是很重要的。

任務：成績處理練習 II

測試資料一

步驟	資料/預期結果
使用者輸入分數並建立分數列表	[-1, 0, 100, 50, 101]
過濾一：去除非法分數	[0, 100, 50]
計算平均值	50
計算標準差	40.825
過濾二：去除離群值	[0, 100, 50]
排序	[0, 50, 100]
列印最低、中位數、最高	0 50 100

任務：成績處理練習 II

測試資料二

步驟	資料/預期結果
使用者輸入分數並建立分數列表	輸入十個 1 和一個 50
過濾一：去除非法分數	[1, 1, …. 50]
計算平均值	5.455
計算標準差	14.087
過濾二：去除離群值	[1, 1, …, 1]
排序	[1, 1, …, 1]
列印最低、中位數、最高	1 1 1

任務：成績處理練習 II

精益求精

- 是否善用內建統計函式 `sum()` 和 `len()`？
- 是否善用 list comprehension?
- 是否考慮特殊情況（例如無輸入或全非法值）？
- 程式可讀性是否夠高？
  - 變數名稱是否有意義
  - 適當處可加上簡短註解（以 # 開頭）

任務：成績處理練習 II

參考答案

```
#輸入分數
scores = [int(e) for e in input().split()]
print('raw: ', scores)

#過濾非法值
scores = [e for e in scores if 0<=e<=100]
print('filter1: ', scores)

size = len(scores)
if size:
    #過濾離群值
    mean = sum(scores)/size
    stdev = (sum([(e - mean)**2 for e in scores])/size)**0.5
    scores = [e for e in scores if mean-3*stdev<=e<=mean+3*stdev]
    print('mean: ', mean, 'stdev: ', stdev)
    print('filter2: ', scores)

    #輸出最低、中位數和最高
    scores.sort()
    size=len(scores)
    if size%2==1:
        print(scores[0], scores[size//2], scores[size-1])
    else:
        print(scores[0], (scores[size//2-1]+scores[size//2])/2, scores[size-1])
```

$$SD = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

成績資料庫

初學者的統計 (III): 列表切片、走訪與搜尋

列表的切片 (slice)

回顧下標運算子與索引值

- 單一索引值可以取出單一元素。

```
data = [8, 4, 20, 99]
print(data[0], data[-1])
data[-2] = 7
print(data)
```

```
8 99
[8, 4, 7, 99]
```

- 使用一組索引值，可以複製出列表的某一段。
  - 注意：不會取到切片結束索引值的那個元素。

```
data = [8, 4, 20, 99]
print(data[1:3])
c = data[0:2]
c[0] = 30
print(data, c)
```

```
[4, 20]
[8, 4, 20, 99] [30, 4]
```

列表的切片 (slice)

省略切片索引值

- 省略起始索引值

```
data = [8, 4, 20, 99]
print(data[:3])
```

```
[8, 4, 20]
```

- 省略結束索引值

```
data = [8, 4, 20, 99]
print(data[2:])
print(data[-3:])
```

```
[20, 99]
[4, 20, 99]
```

搭配 sorted() 或列表的 sort() 操作，
這是取得最高或最低 k 個元素的標準技法。

列表的切片 (slice)

索引變化值

- 每隔 k 個取出

標準技法。

```
data = [11, 22, 33, 44, 55, 66, 77, 88]
print(data[::2])
print(data[1::2])
print(data[1:6:2])
print(data[1:7:2])
```

```
[11, 33, 55, 77]
[22, 44, 66, 88]
[22, 44, 66]
[22, 44, 66]
```

- 也可以倒數回來

反轉列表的標準技法。

```
data = [11, 22, 33, 44, 55, 66, 77, 88]
print(data[::-1])
print(data[::-2])
print(data[2:6:-1])
print(data[6:2:-1])
print(data[-1:-5:-2])
```

```
[88, 77, 66, 55, 44, 33, 22, 11]
[88, 66, 44, 22]
[]
[77, 66, 55, 44]
[88, 66]
```

Snakify 線上練習時間

- 請完成 Snakify 平台主題 **7 Lists** 的習題1 **Even indices**。

穩紮穩打的程式員

```
i = 0
data = input().split()
for e in data:
    if i%2==0:
        print(e)
    i += 1
```

帥氣的 Python 程式員

```
data = input().split()
for e in data[::2]:
    print(e)
```

```
for e in input().split()[::2]:
    print(e)
```

以索引值走訪：range()

- 請完成 Snakify 平台主題
  7 Lists 的習題3 **Greater than previous**。

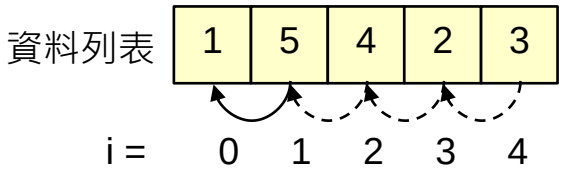
```
data = [int(e) for e in input().split()

if data[1]>data[0]:
   print(data[1])
if data[2]>data[1]:
   print(data[2])
if data[3]>data[2]:
   print(data[3])
if data[4]>data[3]:
   print(data[4])
...
```

運算思維：模式辨識
程式設計：迴圈控制

```
data = [int(e) for e in input().split()]

i = 0
for _ in data:
   if i>0 and data[i]>data[i-1]:
      print(data[i])
   i += 1
```



以索引值走訪：range()

- 當我們需要以索引值走訪列表時，如何簡便地產生 0, 1, 2, 3, … 的索引值？

手工產生（苦哈哈且不實際）

```
data = [int(e) for e in input().split()]

for i in [0, 1, 2, 3, 4]:
    print(data[i])
```

```
10 20 30 40 55
10
20
30
40
55
```

使用 range() 產生

```
data = [int(e) for e in input().split()]

for i in range(5):
    print(data[i])
```

```
10 20 30 40 55
10
20
30
40
55
```

以索引值走訪：range()

- range() 能為我們快速產生一連串規律變化的整數值。

```
for i in range(5):
    print(i, end=' ')
print()

for i in range(3, 7):
    print(i, end=' ')
print()

for i in range(10, 21, 2):
    print(i, end=' ')
print()

for i in range(-10, 0, 1):
    print(i, end=' ')
print()
```

```
0 1 2 3 4
3 4 5 6
10 12 14 16 18 20
-10 -9 -8 -7 -6 -5 -4 -3 -2 -1
```

range() 函式並非一口氣回傳一個列表，但在此我們就先略過不談這些細節了。

以索引值走訪：range()

請完成 snakify 平台主題

7 Lists 的習題3 **Greater than previous**。

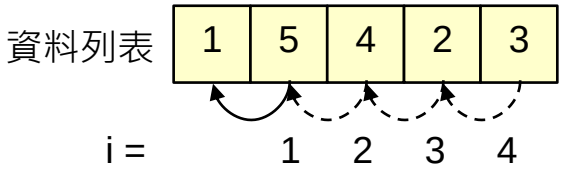
```
data = [int(e) for e in input().split()]
i = 0
for _ in data:
    if i>0 and data[i]>data[i-1]:
        print(data[i])
    i += 1
```

熟悉迴圈和列表的程式員

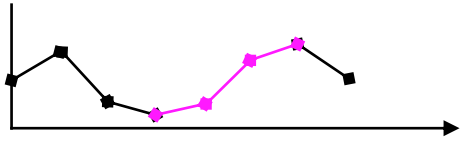
```
data = [int(e) for e in input().split()]

for i in range( 1, len(data)):
    if data[i]>data[i-1]:
        print(data[i])
```

熟悉 range() 的程式員



可擴充本程式以觀察一段時間資料的上升模式。
例如可計算學生成績的連續進步次數。



以索引值走訪：range()

Snakify 平台主題 7 Lists

- 習題3 **Greater than previous**。

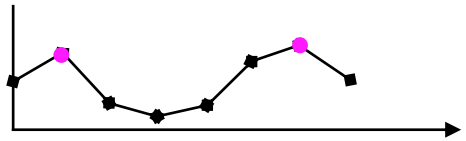
資料列表

1	5	4	2	3

i = 1 2 3 4 → `range(1, len(data))`

- 習題5 **Greater than neighbors**。

本程式可觀察一段資料的峰值。



資料列表

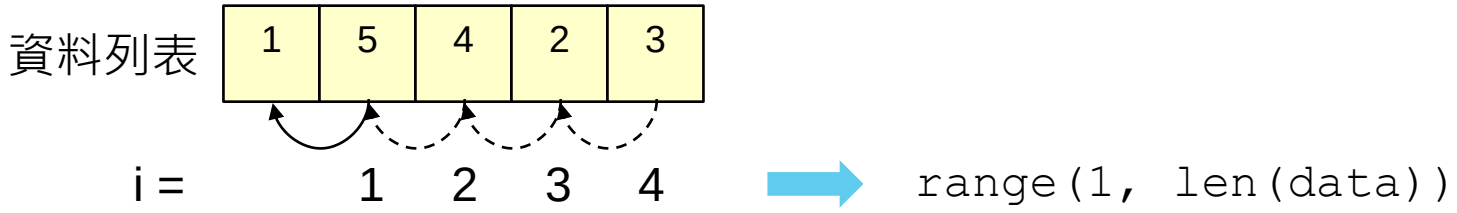
1	5	4	2	3

i = 1 2 3 → `range(1, len(data)-1)`

以索引值走訪：range()

Snakify 平台主題 7 Lists

- 習題4 **Neighbors of the same sign**。



兩個小任務

- 小任務一：如何判斷同號？
- 小任務二：只印出第一組相鄰且同號的資料。

以索引值走訪：range()

Snakify 平台主題 7 Lists

- 習題4 **Neighbors of the same sign**。

```
found = False
n = [int(e) for e in input().split()]
for i in range(1, len(n)):
    if not found and n[i-1]*n[i]>0:
        print(n[i-1], n[i])
        found = True
```

使用布林變數來記錄與判斷是否要列印。

缺點：寫作麻煩，而且執行時效率較差。

```
n = [int(e) for e in input().split()]
for i in range(1, len(n)):
    if n[i-1]*n[i]>0:
        print(n[i-1], n[i])
        break
```

使用 break 敘述。

在迴圈中，遇到 break 便立即中斷且跳出迴圈。

以索引值走訪：range()

Snakify 平台主題 7 Lists

- 習題7 **The number of distinct numbers**。

思考點

- 如果資料已經排序了，怎麼樣計算相異數值的個數呢？
- 如果我們一路看過去，怎麼樣叫作發現一個新的數值？

資料列表

1	2	2	3	3	3

i = 1 2 3 4 5 → `range(1, len(data))`

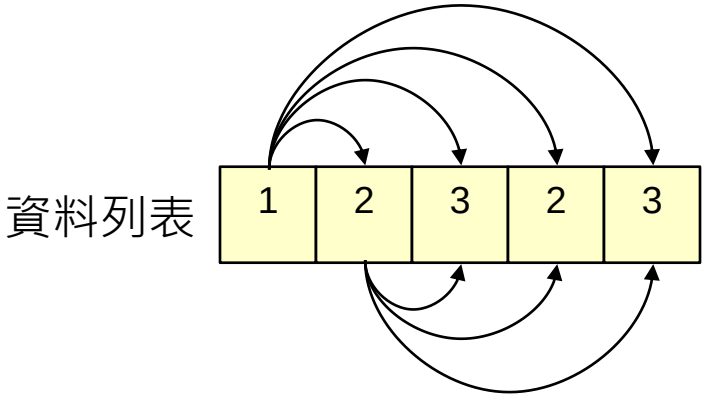
不管資料有沒有排序，這個問題，帥氣的 Python 程式員會這樣寫：

```
print(len(set(input().split())))
```

以索引值走訪：range()

Snakify 平台主題 7 Lists

- 習題10 **The number of pairs of equal**。



- 只要想得到用兩層迴圈，你就突破盲腸了！

```
for i in range(        ):
    for j in range(         ):
        if      ==      :
```

以索引值走訪：range()

- range() 的另一個用途是重覆執行指定次數。

範例：很重要所以說三遍。

```
for _ in range(3):
    print('很重要所以說三遍')
```

```
很重要所以說三遍
很重要所以說三遍
很重要所以說三遍
```

這裡你當然也可以利用字串的乘法運算來完成。

使用 _ 作為變數名稱暗示沒有要使用到 0, 1, 2 這三個值。

範例：輸入四位同學的平時成績並作統計

```
for _ in range(4):
    scores = [int(e) for e in input('輸入成績>').split()]
    if scores:
        print('有', len(scores), '個成績，平均',
              round(sum(scores)/len(scores)))
    else:
        print('沒有成績')
```

```
輸入成績>100 90
有 2 個成績，平均 95
輸入成績>
沒有成績
輸入成績>100
有 1 個成績，平均 100
輸入成績>100 90 80 100
有 4 個成績，平均 92
```

分解賦值 (Unpack)

- 列表有一個很有趣的分解賦值操作，讓我們可以很方便地用具名變數指稱其中的每個元素。

```
x, y = [10, 20]
print(x, y)
```

```
10 20
```

```
data = [10, 20]
print(data)
x, y = data
print(x, y)
```

```
[10, 20]
10 20
```

```
a, b, c = [int(e) for e in input('請輸入三個整數>').split()]
print(a, b, c)
```

```
請輸入三個整數>19 88 3
19 88 3
```

```
a, b = [10]          #ValueError: not enough values to unpack (expected 2, got 1)
a, b = [10, 20, 30]  #ValueError: too many values to unpack (expected 2)
```

分解賦值 (Unpack)

- 交換兩個變數是一個常見操作，Python 的標準寫法是

```
x, y = [10, 20]
print(x, y)
x, y = y, x
print(x, y)
```

```
10 20
20 10
```

- 這裡的交換其實就是在作分解賦值，只不過是對元組 (tuple) 作。
  - 元組型態和列表型態相似，但是元組是不可修改的。

```
x = 10
y = 20
print(x, y)
t = y, x
print(type(t), t)
x, y = t
print(x, y)
```

```
10 20
<class 'tuple'> (20, 10)
20 10
```

```
t = (17, 21, 33)
for e in t:
    print(e)
print(t[1])
t[1] = 3
```

```
17
21
33
21
```

分解賦值 (Unpack)

- Snakify 平台主題 **7 Lists**
  - 習題8 **Swap neighbors**。

資料列表

1	2	2	3	3	3

i = 0 1 2 3 4 5 → `range(0, len(data), 2)`

列表的搜尋

Snakify 平台主題 7 Lists

- 習題9 **Swap min and max**。
  - 步驟一：找到 min 和 max 的位置（索引值），這個我們已經在習題6 **The largest element** 作過了。
  - 步驟二：使用分解賦值來交換。

作法一

```
min_ind = max_ind = 0
data = [int(e) for e in input().split()]
for i in range(1, len(data)):
    if data[i]> data[max_ind]:
        max_ind = i
    elif data[i]< data[min_ind]:
        min_ind = i

data[max_ind], data[min_ind] = data[min_ind], data[max_ind]
```

列表的搜尋

Snakify 平台主題 **7 Lists**

- 習題9 **Swap min and max**。
  - 除了自己寫程式走訪列表資料以找到最小/大值的位置外，列表有個方便的操作稱為 `index()`。
  - `lst.index(e)` 會回傳 `lst` 列表中第一次出現的 `e` 的索引值。

```
data = [10, 20, 88, 45, 88]
ind = data.index(88)
print(ind)
data[ind] = 99
print(data)
```

```
2
[10, 20, 99, 45, 88]
```

列表的搜尋

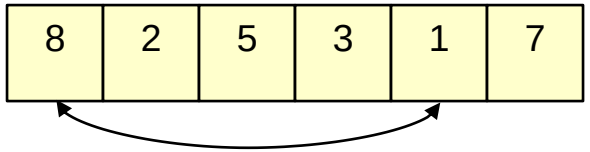
Snakify 平台主題 7 Lists

- 習題9 **Swap min and max**。
  - 使用內建函式 `min()`, `max()`, 列表的 `index()` 和分解賦值，以 Python 風完成本題。

作法二

```
data = [int(e) for e in input().split()]
a = data.index(max(data))
b = data.index(min(data))
data[a], data[b] = data[b], data[a]
```

資料列表



max(n) 是 8
index(8) 是 0
min(n) 是 1
index(1) 是 4

這個寫法雖然簡潔，但是和前前頁自行撰寫 for 迴圈的程式比起來，
這個程式會掃過資料四次，但前前頁的程式只會掃過一次。

列表的搜尋

- 列表的 index() 操作

```
data = [int(e) for e in input().split()]
key = 3
```

```
pos = data.index(3)
```

```
for i in range(len(data)):
    if data[i]==key:
        pos = i
        break
```

index() 進行線性搜尋，找到列表中的第一個指定值，如同右邊的程式碼作的工作。

列表的搜尋

列表的 in 運算

- 若單純想判斷資料是否存在於列表中，可以使用 in 運算。
- 為避免 index() 搜尋不存在的資料發生錯誤，可先用 in 運算。

```
data = [int(e) for e in input().split()]
print(data.index(3))
```

```
1 2 4
Traceback (most recent call last):
ValueError: 3 is not in list
```

```
for _ in range(2):
    data = [int(e) for e in input().split()]
    key = 3
    if key in data:
        print(key, '的位置在', data.index(key))
    else:
        print(key, '不存在')
```

```
1 2 4
3 不存在
1 2 4 5 3
3 的位置在 4
```

```
scores = [int(e) for e in input().split()]
if 100 not in data:
    print('沒有人滿分')
```

想檢查不存在，就加個 not。

Snakify 線上練習平台：作業三

- 請完成 Snakify 平台主題 **7 Lists** 的習題。

- 計分方式（以題數計）：
  - 第 1-4 題，每題 +20%
  - 第 5-13 題，每題 +5%

答對任四題：80分
答對任六題：90分

- 提示
  - 課堂上已示範五題 (1, 2, 3, 4, 6)，至少 85 分了。☺

成績資料庫

初學者的統計 (IV): 多維度列表

多維度列表

- 還記得我們講過列表的 append() 和 + 運算嗎？

自建列表與內建函式

- 目標：從字串列表建立整數列表

列表的 append() vs. + 運算子

```
a = [1, 2, 3]
b = [4, 5]

a.append(b)
print(a)
print(b)
```

```
[1, 2, 3, [4, 5]]
[4, 5]
```

```
a = [1, 2, 3]
b = [4, 5]

a = a + b # 也可以寫成 a+=b
print(a)
print(b)
```

```
[1, 2, 3, 4, 5]
[4, 5]
```

推動大學程式設計教學計畫。分項六：資料分析領域教學研發推廣團隊

29

多維度列表

- 假如今天我們要讀入四個學生的成績，每人三個成績，你喜歡哪一種寫法呢？

寫法一

```
scores = []
for i in range(4):
    scores.append([int(e) for e in input().split()])
print(scores)
print(scores[1][2])  #第二位學生的第三個成績
```

```
100 90 80
90 90 70
60 80 95
100 100 60
[[100, 90, 80], [90, 90, 70], [60, 80, 95], [100, 100, 60]]
70
```

寫法二

```
scores = []
for i in range(4):
    scores += [int(e) for e in input().split()]
print(scores)
print(scores[1*3+2])  #第二位學生的第三個成績
```

如果我們寫成 `[[int(e) … ]]`，也就是兩層 `[]`，也能有和上面相同的效果。

```
100 90 80
90 90 70
60 80 95
100 100 60
[100, 90, 80, 90, 90, 70, 60, 80, 95, 100, 100, 60]
70
```

多維度列表

- 我們能以一次或兩次下標運算來取用二維列表中的元素，端看你想要取得什麼。

二維列表：列表中的元素又是列表。

```
scores = []
for _ in range(4):
    scores.append([int(e) for e in input().split()])
print(scores)

print('第一位學生的平均分數 = ', sum(scores[0])/len(scores[0]))

i = 1
j = 2
print('第', i+1, '位學生的第', j+1, '個成績 =', scores[i][j])
```

```
100 90 80
90 90 70
60 80 95
100 100 60
[[100, 90, 80], [90, 90, 70], [60, 80, 95], [100, 100, 60]]
第一位學生的平均分數 =  90.0
第 2 位學生的第 3 個成績 = 70
```

擁抱 list comprehension 的程式員會這樣寫：

```
scores = [ [int(e) for e in input().split()] for _ in range(4)]
```

多維度列表

以 for 迴圈走訪列表

一層迴圈走訪一維列表

```
scores = [int(e) for e in input().split()]
for e in scores:
    print(e, end=' ')
print()
```

```
99 88 77 66 55
99 88 77 66 55
```

兩層迴圈走訪二維列表

```
scores = []
for i in range(4):
    scores.append([int(e) for e in input().split()])

for stu in scores:
    for sc in stu:
        print(sc, end=' ')
    print() #注意：這一行要放在第一層迴圈內
```

```
100 90 80
90 90
70 80 95
100 100 60 80
100 90 80
90 90
70 80 95
100 100 60 80
```

每位學生可以有不同個數的分數。

多維度列表

以 for 迴圈走訪列表

一層迴圈走訪一維列表

```
scores = [int(e) for e in input().split()]
for i in range(len(scores)):
    print(scores[i], end=' ')
print()
```

```
99 88 77 66 55
99 88 77 66 55
```

兩層迴圈走訪二維列表

```
scores = []
for i in range(4):
    scores.append([int(e) for e in input().split()])

for i in range(len(scores)):
    for j in range(len(scores[i])):
        print(scores[i][j], end=' ')
    print()
```

```
100 90 80
90 90
70 80 95
100 100 60 80
100 90 80
90 90
70 80 95
100 100 60 80
```

每位學生可以有不同個數的分數。

Snakify 線上練習時間

- 請完成 Snakify 平台主題

9 Two-dimensional lists 的習題7 **Scale a matrix**。

```
R, C = [int(e) for e in input().split()]
matrix = []
for r in range(R):
    matrix.append([int(e) for e in input().split()])

scale = int(input())
for r in range(R):
    for c in range(C):
        print(matrix[r][c]*scale, end=' ')
    print()
```

總有一天你會了解我，然後愛上我

```
matrix = [[int(e) for e in input().split()] for r in range(R)]
```

Snakify 線上練習時間

- 請完成 Snakify 平台主題

9 Two-dimensional lists 的習題1 **Maximum**。

```
R, C = [int(e) for e in input().split()]
data = []
for i in range(R):
    data.append([int(e) for e in input().split()])


M = data[0][0]
Mr = Mc = 0   #最大值的列索引值和行索引值

for r in range(R):
    for c in range(C):
        if data[r][c]>M: #假如 data[r][c] 比現有最大值還大
            M = data[r][c]
            Mr, Mc = r, c

print(Mr, Mc)
```

總有一天你會了解我，然後愛上我

```
data = [[int(e) for e in input().split()] for r in range(R)]
```

多維度列表

習慣 C/C++ 語言的人，可能會想知道如何預先定義出一、二維列表。

```
data = [0]*10
print(data)
data = [0 for _ in range(10)]
print(data)
```

```
int data[10]={};
```

```
data = [[0]*3 for _ in range(4)]
data[0][0] = 3
print(data)
```

```
int data[4][3]={};
```

```
[[3, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0]]
```

```
data = [[i for i in range(3)] for j in range(4)]
print(data)

data = [[i+j*3 for i in range(3)] for j in range(4)]
print(data)
```

```
[[0, 1, 2], [0, 1, 2], [0, 1, 2], [0, 1, 2]]
```

```
[[0, 1, 2], [3, 4, 5], [6, 7, 8], [9, 10, 11]]
```

```
data = [[0]*3]*4
data[0][0] = 3
print(data)
```

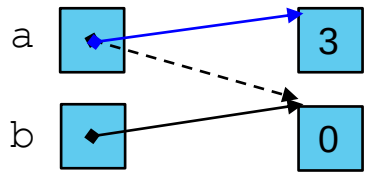
```
[[3, 0, 0], [3, 0, 0], [3, 0, 0], [3, 0, 0]]
```

想像 data 中的四個元素「指向」同一個列表。

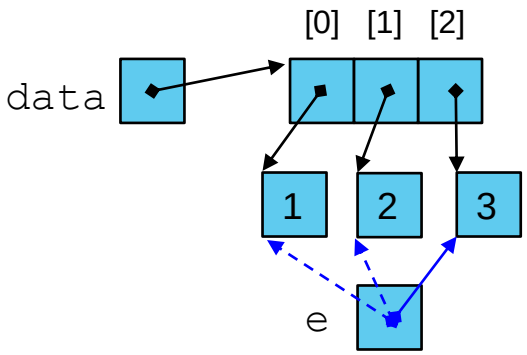
Python 的指派意義

- a=b 的意思是 a 指稱/代表 b（所指稱/代表的資料）。

```
a = 0
b = 0
a = 3
```




```
data = [1, 2, 3]
for e in data:
    print(e)
```

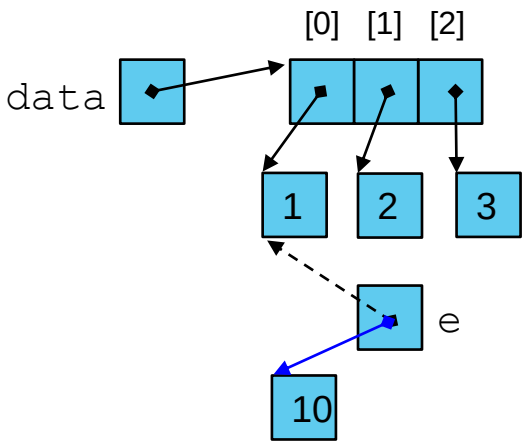


Python 的指派意義

a=b 的意思是 a 指稱/代表 b（所指稱/代表的資料）。

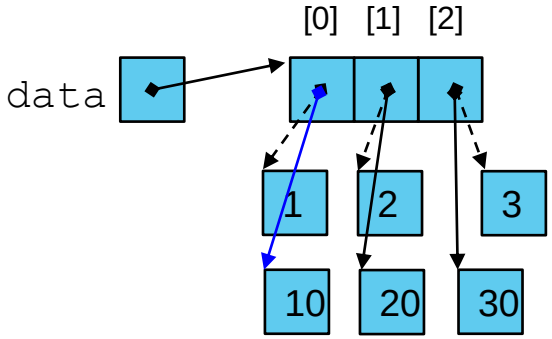
```
data = [1, 2, 3]
for e in data:
    print(e)
    e = e*10
    print(e)
print(data)
```

```
1
10
2
20
3
30
[1, 2, 3]
```




```
data = [1, 2, 3]
for i in range(len(data)):
    print(data[i])
    data[i] = data[i]*10
    print(data[i])
print(data)
```

```
1
10
2
20
3
30
[10, 20, 30]
```



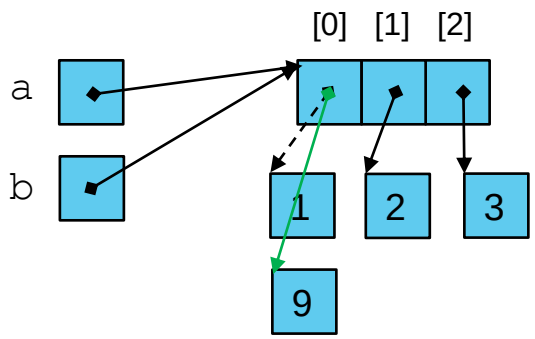
實務技法：要直接修改列表資料時，請以索引值搭配下標運算取用元素。

Python 的指派意義

a=b 的意思是 a 指稱/代表 b（所指稱/代表的資料）。

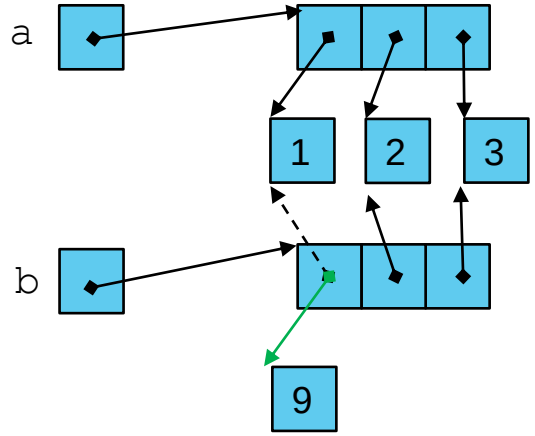
```
a = [1, 2, 3]
b = a
b[0] = 9
print(a, b)
```

```
[9, 2, 3] [9, 2, 3]
```




```
a = [1, 2, 3]
b = a[:]
b[0] = 9
print(a, b)
```

```
[1, 2, 3] [9, 2, 3]
```



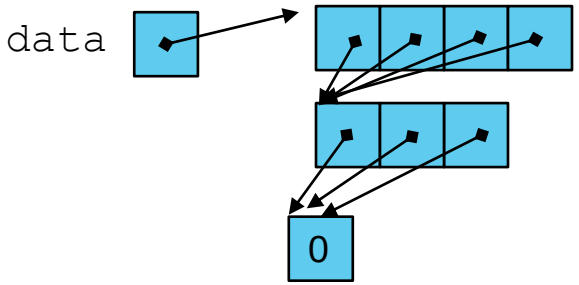
實務技法：複製列表時使用 [:]。

Python 的指派意義

a=b 的意思是 a 指稱/代表 b（所指稱/代表的資料）。

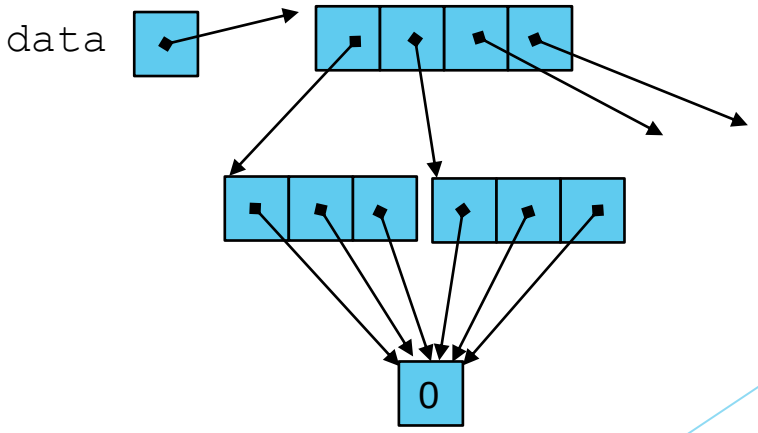
```
data = [[0]*3]*4
data[0][0] = 3
print(data)
```

[[3, 0, 0], [3, 0, 0], [3, 0, 0], [3, 0, 0]]




```
data = [[0]*3 for _ in range(4)]
data[0][0] = 3
print(data)
```

[[3, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0]]



實務技法：定義多維列表時避免使用 * 運算。

任務：成績處理練習 III

計算平均分數

- 運用所學，撰寫程式使其具備以下功能：
  - 讀入五位學生的成績（成績數量不定），每位學生的成績在一行。
  - 若某位學生的成績數量少於四次，不足的成績補 0 分。
  - 若某位學生的成績數量多於四次，只採計最高分的四次。
  - 請輸出五位學生的四次成績（由高到低）和平均分數，以二維表格呈現。

範例一

```
100 90 80 70
60 50 40 30
30 30 30 30
20 30 40 50
91 92 93 94
100 90 80 70 85.0
60 50 40 30 45.0
30 30 30 30 30.0
50 40 30 20 35.0
94 93 92 91 92.5
```

範例二

```
100 90 80 70
80 80 80 80
100 100
100 0 10 20 90 90 100
我是空白行-第五位學生很不認真
100 90 80 70 85.0
80 80 80 80 80.0
100 100 0 0 50.0
100 100 90 90 95.0
0 0 0 0 0.0
```

任務：成績處理練習 III

計算平均分數

- 運用所學，撰寫程式使其具備以下功能：
  - 讀入五位學生的成績（成績數量不定），每位學生的成績在一行。
  - 若某位學生的成績數量少於四次，不足的成績補 0 分。
  - 若某位學生的成績數量多於四次，只採計最高分的四次。
  - 請輸出五位學生的四次成績（由高到低）和平均分數，以二維表格呈現。
- 思考程式流程

目標	程式
讀入五位學生的成績	• 使用 `for` 句型以及列表的 `append()`。
成績數量少於四次，補 0 分。	• 使用 `len()` 取得成績數量。 • 使用列表的 `append()` 或運算子 + 來補 0 分。
成績數量多於四次，採計最高四次。	• 使用 `sorted()` 或列表的 `sort()` 函式。 • 使用列表的切片取出最高四次。
輸出成績和平均	• 使用雙層 `for` 迴圈。 • 使用 `sum()` 和 `len()` 計算平均。

任務：成績處理練習 III

計算平均分數

```
data = []
for i in range(5):
    data.append([int(e) for e in input().split()])

NumCountedScores = 4

for stu in data: #對每位學生成績 stu
    #將 stu 由小到大排序，反轉，取出前 NCS 個
    stu = sorted(stu)[::-1][:NumCountedScores]
    #不足 NCS 個分數補 0 分
    if len(stu)<NumCountedScores:
        stu += [0]*(NumCountedScores-len(stu))
    #列印分數與平均
    for sc in stu:
        print(sc, end=' ')
    print(sum(stu)/len(stu))
print()
```

```
100 90 80 70
80 80 80 80
100 100
100 0 10 20 90 90 100
我是空白行-第五位學生很不認真
100 90 80 70 85.0
80 80 80 80 80.0
100 100 0 0 50.0
100 100 90 90 95.0
0 0 0 0 0.0
```

排序的部份也可以利用 sorted() 的參數 reverse 來由大到小排序，sorted(stu, reverse=True)。
另外，如果不要保留原始資料的順序，也可以直接排序，stu.sort(reverse=True)。

任務：成績處理練習 III

計算平均分數

```
data = []
for i in range(5):
    data.append([int(e) for e in input().split()])

NumCountedScores = 4

for stu in data: #對每位學生成績 stu
    #補進至少 NCS 個 0 分
    stu += [0]*NumCountedScores
    #將 stu 由小到大排序，反轉，取出前 NCS 個
    stu = sorted(stu)[::-1][:NumCountedScores]
    #列印分數與平均
    for sc in stu:
        print(sc, end=' ')
    print(sum(stu)/len(stu))
print()
```

```
100 90 80 70
80 80 80 80
100 100
100 0 10 20 90 90 100
我是空白行-第五位學生很不認真
100 90 80 70 85.0
80 80 80 80 80.0
100 100 0 0 50.0
100 100 90 90 95.0
0 0 0 0 0.0
```

處理不足分數的另一種作法：預先補進 NCS 個 0 分。

任務：成績處理練習 IV

自己的分數自己算之「計算 snakify 作業得分」

- 運用所學，撰寫程式使其具備以下功能：
  - 讀入三位學生的 snakify 答題時間紀錄，每位學生的紀錄在一行。
  - 前四題（最早答對的四題）每題得分 20 分，接下去每題得分 5 分。
  - 紀錄中的數值若為負，表示遲交，分數打 8 折。
  - 計算並列印每位學生的總得分。

範例一

```
張三 10 20 90 40
李四 1 2 3 4 5
王五 3 4 -1 4 5
張三 80
李四 85
王五 84
```

範例二

```
張三 10 20 90
李四 1 2 3 4 5 6
王五 3 4 -1 -1 10 2
張三 60
李四 90
王五 88
```

任務：成績處理練習 IV

自己的分數自己算之「計算 snakify 作業得分」

- 運用所學，撰寫程式使其具備以下功能：
  - 前四題每題得分 20 分，接下去每題得分 5 分。
  - 紀錄中的數值若為負，表示遲交，分數打 8 分。

範例一

```
張三 10 20 90 40
李四 1 2 3 4 5
王五 3 4 -1 4 5
張三 80
李四 85
王五 84
```

思考處理流程

看過每個時間戳記
　如果是前四題，算 20 分；
　其它的，算 5 分。
　如果是遲交 (負值)，打八折。

張三

	[10,	20,	90,	40]
i	0	1	2	3
得分	20	20	20	20
總分	20	40	60	**80**

李四

	[1,	2,	3,	4,	5]
i	0	1	2	3	4
得分	20	20	20	20	5
總分	20	40	60	80	**85**

好像有點感覺了！

任務：成績處理練習 IV

自己的分數自己算之「計算 snakify 作業得分」

- 運用所學，撰寫程式使其具備以下功能：
  - 前四題每題得分 20 分，接下去每題得分 5 分。
  - 紀錄中的數值若為負，表示遲交，分數打 8 分。

範例一

```
張三 10 20 90 40
李四 1 2 3 4 5
王五 3 4 -1 4 5
張三 80
李四 85
王五 84
```

思考處理流程

看過每個時間戳記
　如果是前四題，算 20 分；
　其它的，算 5 分。
　如果是遲交 (負值)，打八折。

```
total = 0
for i in range(len(scores)):
    s = 20
    if i>=4: #前四題 20 分，第五題開始 5 分
        s = 5
    if scores[i]<0: #遲交打八折
        s *= 0.8
    total += s
```

任務：成績處理練習 IV

自己的分數自己算之「計算 snakify 作業得分」

- 運用所學，撰寫程式使其具備以下功能：
  - 前四題每題得分 20 分，接下去每題得分 5 分。
  - 紀錄中的數值若為負，表示遲交，分數打 8 分。

範例一

```
張三 10 20 90 40
李四 1 2 3 4 5
王五 3 4 -1 4 5
張三 80
李四 85
王五 84
```

思考處理流程

看過每個時間戳記
　如果是前四題，算 20 分；
　其它的，算 5 分。
　如果是遲交 (負值)，打八折。

王五

```
     [3,    4,   -1,    4,    5]

i     0     1     2     3     4
得分 20    20    20    20     5
打折 20    20    16    20     4
總分 20    40    56    76    80
```

王五

```
     [3,    4,    4,    5,   -1]

i     0     1     2     3     4
得分 20    20    20    20     5
打折 20    20    20    20     4
總分 20    40    60    80    84
```

把遲交的習題放在前四題好像很吃虧，一次就損失4分。
我們應該把遲交的習題放在後面計分。⇒ 由大到小排序。

任務：成績處理練習 IV

自己的分數自己算之「計算 snakify 作業得分」

```
data = []
for i in range(3): #每一行代表學生資料（姓名與每題繳交時間）
    stu = input().split()
    stu[1:] = [int(e) for e in stu[1:]]
    data.append(stu)

for stu in data:
    scores = sorted(stu[1:])[::-1] #將時間由大到小排序
    total = 0
    for i in range(len(scores)):
        s = 20
        if i>=4: #前四題 20 分，第五題開始 5 分
            s = 5
        if scores[i]<0: #遲交打八折
            s *= 0.8
        total += s
    stu.append(round(total)) #順手把得分加進去
    print(stu[0], stu[-1])
print()
```

計算分數

```
張三 10 20 90 40
李四 1 2 3 4 5
王五 3 4 -1 4 5
張三 80
李四 85
王五 84
```

data

```
[
['張三', 10, 20, 90, 40],
['李四', 1, 2, 3, 4, 5],
['王五', 3, 4, -1, 4, 5]
]
```

```
[
['張三', 10, 20, 90, 40, 80],
['李四', 1, 2, 3, 4, 5, 85],
['王五', 3, 4, -1, 4, 5, 84]
]
```

任務：成績處理練習 IV

自己的分數自己算之「計算 snakify 作業得分」

```
NumStudents = 3

#前四題 20 分，第五題開始 5 分
NumHighScoreProblems = 4
HighScore = 20
LowScore = 5

#遲交打八折
DelayPenalty = 0.8

data = []
for i in range(NumStudents): #每一行代表每題繳交時間
    line = input().split()
    stu = [line[0]] #姓名
    stu += [int(e) for e in line[1:]]
    data.append(stu)

for stu in data:
    scores = sorted(stu[1:])[::-1] #由大到小排序
    total = 0
    for i in range(len(scores)):
        s = HighScore
        if i>= NumHighScoreProblems :
            s = LowScore
        if scores[i]<0:
            s *= DelayPenalty
        total += s
    stu.append(round(total)) #順手把得分加進去
    print(stu[0], stu[-1])
print()
```

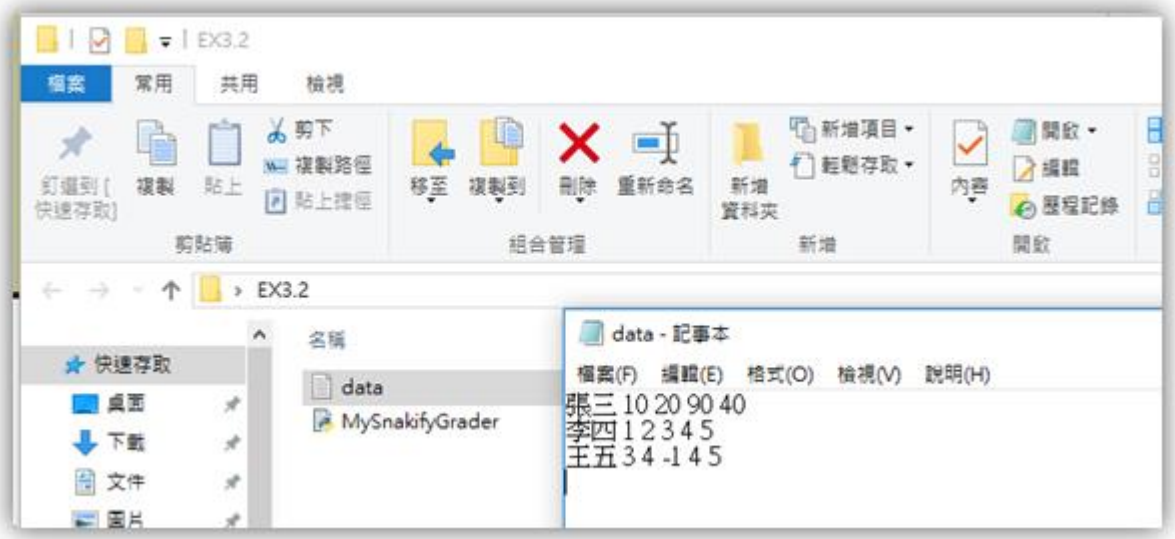
當程式當中有較多人為設定數值時，
建議將它們命名並集中設定。
一來方便閱讀程式，二來方便修改。

任務：成績處理練習 IV

- 自己的分數自己算之「計算 snakify 作業得分」
  - 簡單檔案處理

現在我們把資料存在文字檔 *data.txt* ，該如何使用我們的程式 *MySnakifyGrader.py* 來處理呢？

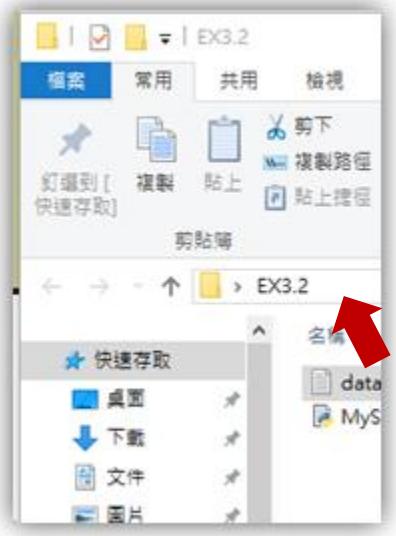
複製貼上啊！



任務：成績處理練習 IV

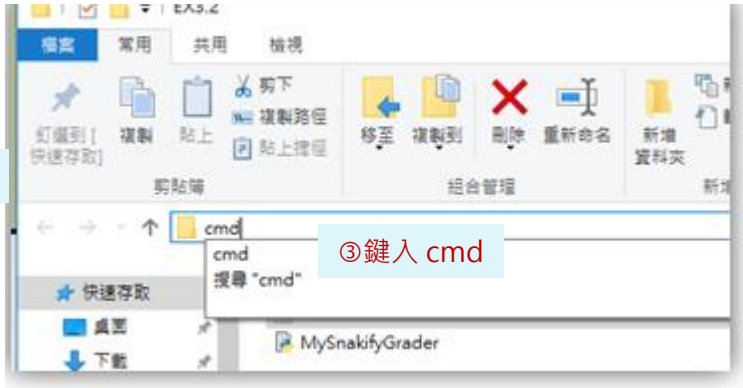
- 自己的分數自己算之「計算 snakify 作業得分」
  - 簡單檔案處理

複製貼上是一種作法，現在我們來學另一種作法。

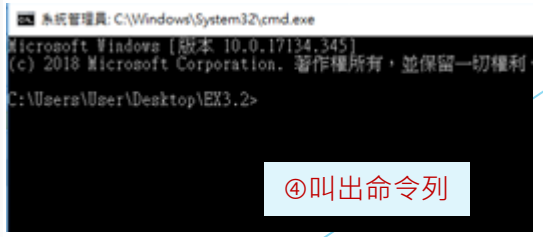


①點旁邊，不要點到目錄名稱

②點成功的話，會變成這樣。



③鍵入 cmd

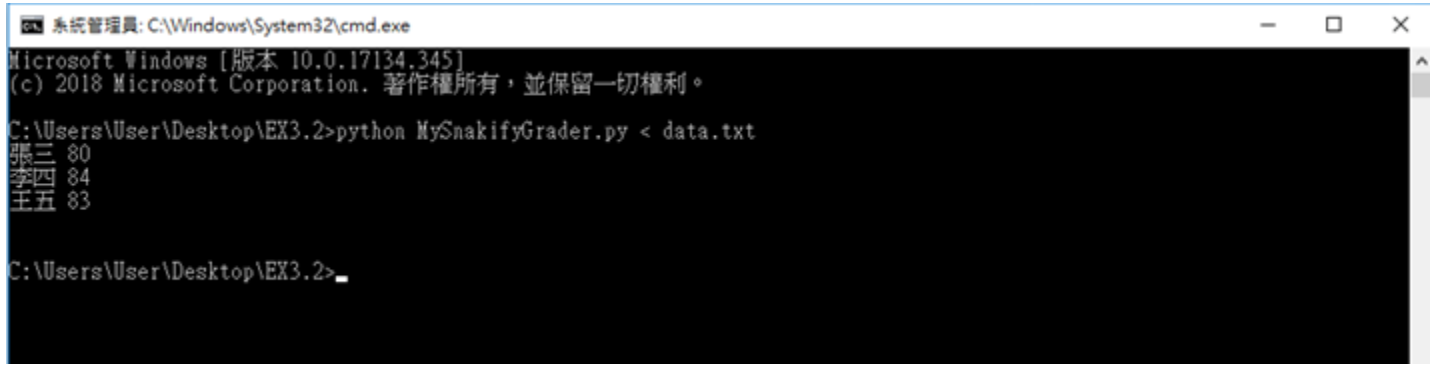


④叫出命令列

任務：成績處理練習 IV

- 自己的分數自己算之「計算 snakify 作業得分」
  - 簡單檔案處理

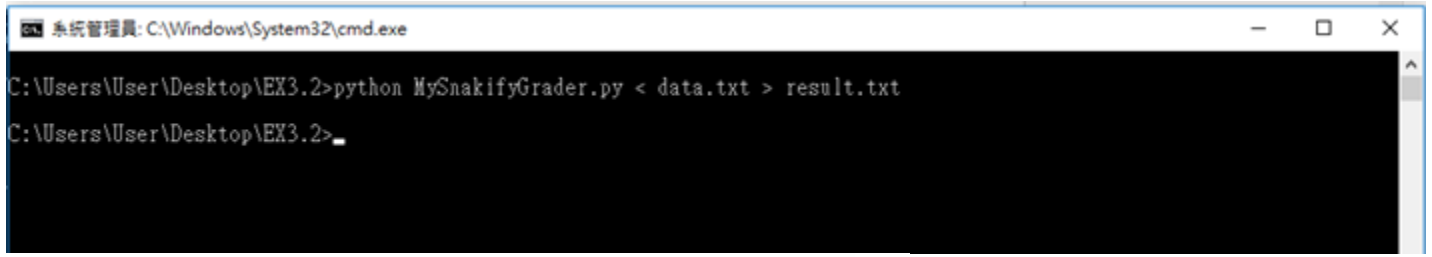
在命令列呼叫 Python 直譯器，將資料檔 *data.txt* 導入作為 *MySnakifyGrader.py* 的輸入。

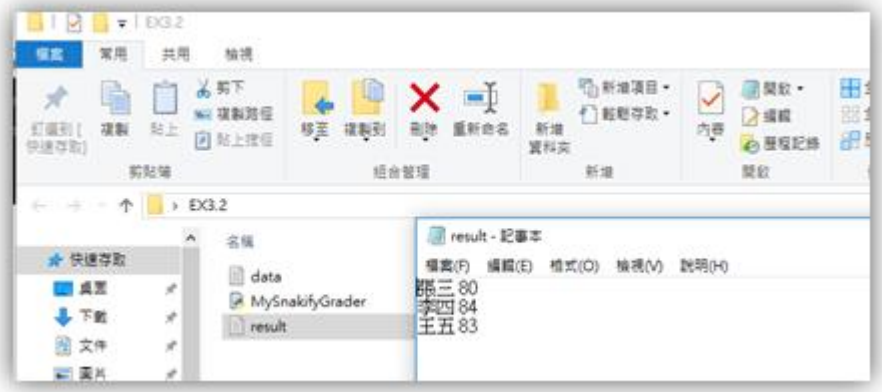


任務：成績處理練習 IV

- 自己的分數自己算之「計算 snakify 作業得分」
  - 簡單檔案處理

我們還可以將輸出結果導出到另一個文字檔 *result.txt* (檔名任取)。





雖然我們還沒有正式教到撰寫程式來處理檔案，但這個小技巧應該能讓你離你的資料分析專題更進一點。☺

Snakify 線上練習平台：作業四

- 請完成 Snakify 平台**主題 9 Two-dimensional lists** 的習題。

- 計分方式（以題數計）：
  - 第 1-2 題，每題 +40%
  - 第 3-8 題，每題 +5%

答對任四題：90分
答對任六題：100分

- 提示
  - 課堂上已示範兩題 (1 & 7)，至少 80 分了。☺
  - **3 Chessboard**：r+c 偶數就是 '.'，奇數就是 '*'。
  - **2 Snowflake**： 第 `n//2` 列、第 `n//2` 行、`r==c`、`r+c==n-1` 這些位置是 `'*'`。